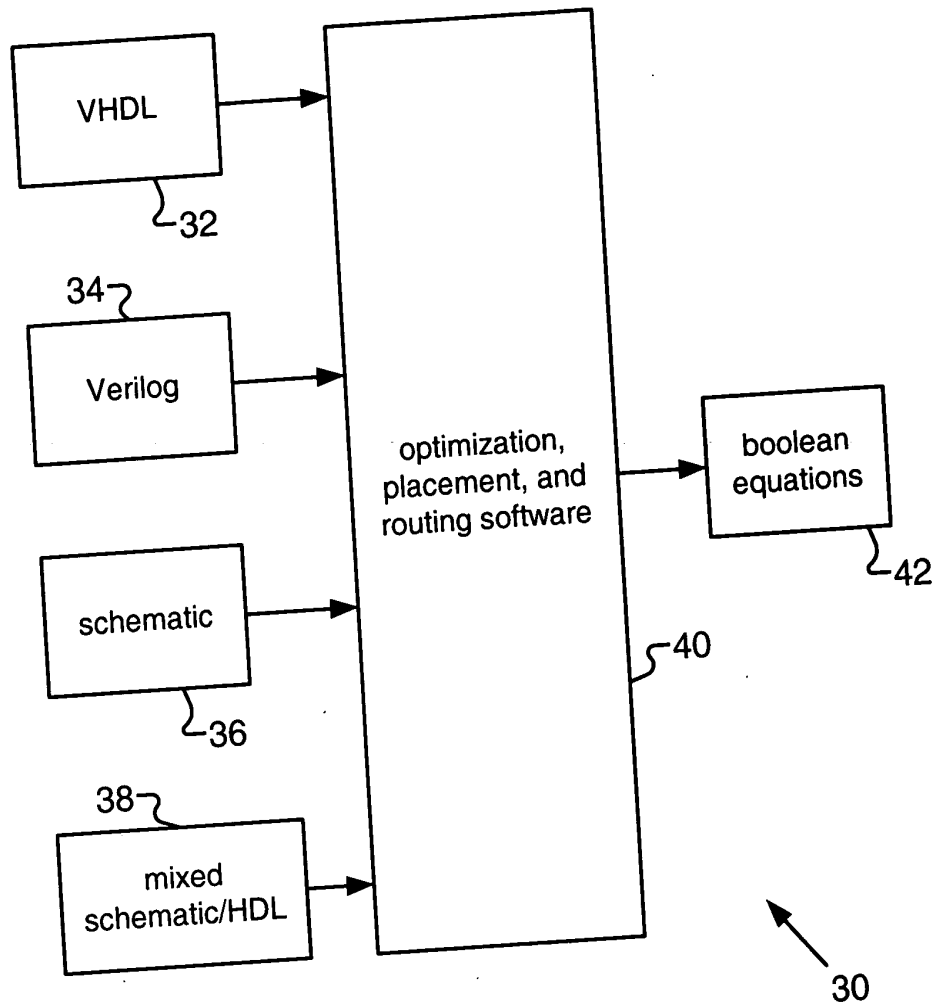


FIG. 1 is a block diagram of a prior art system for generating boolean equations from various input formats.



prior art

FIG. 1

FIG. 2

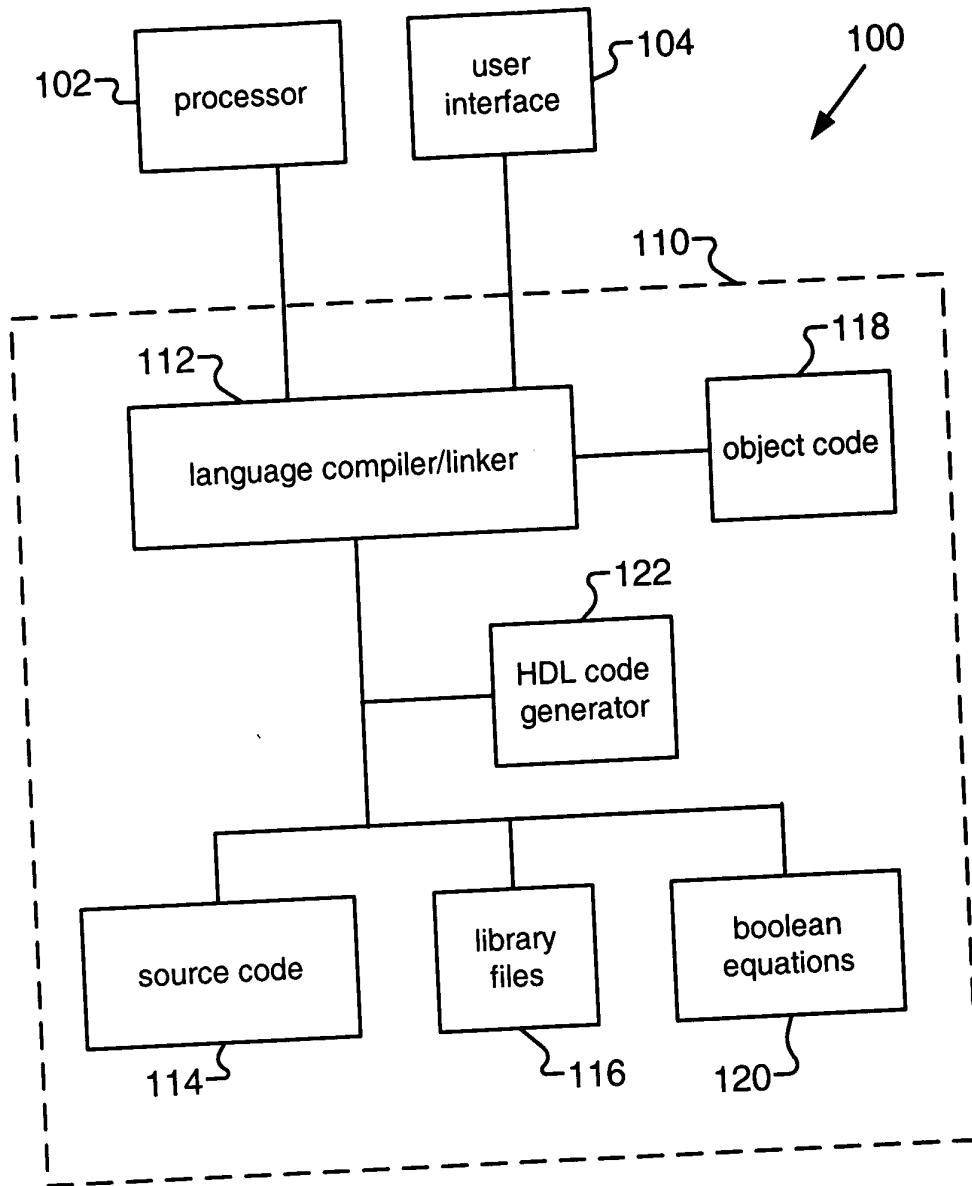


FIG. 2

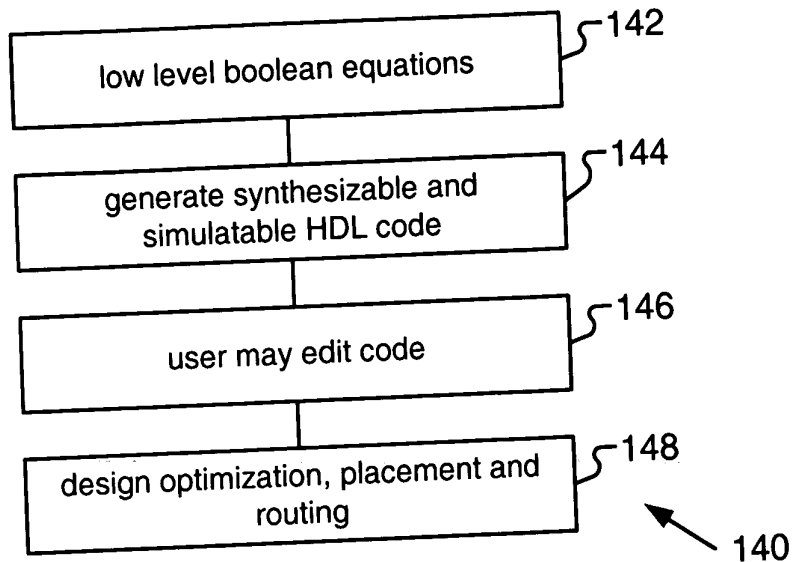


FIG. 3

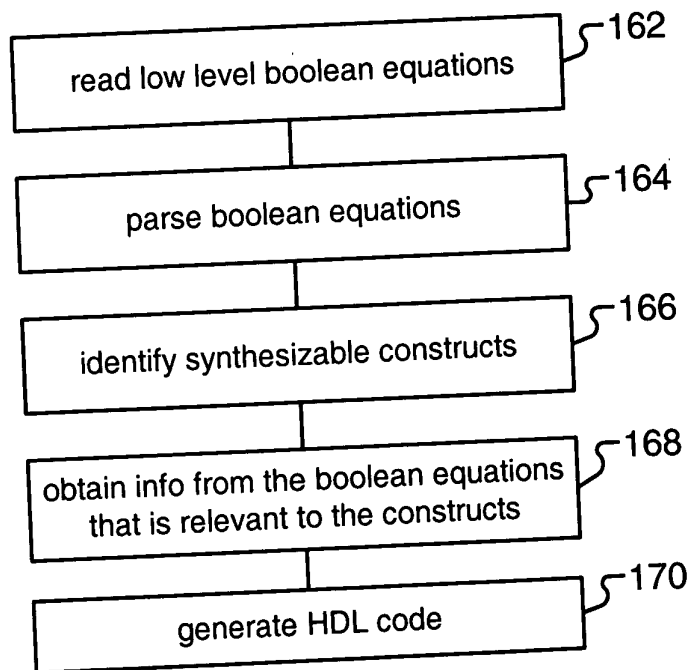


FIG. 4

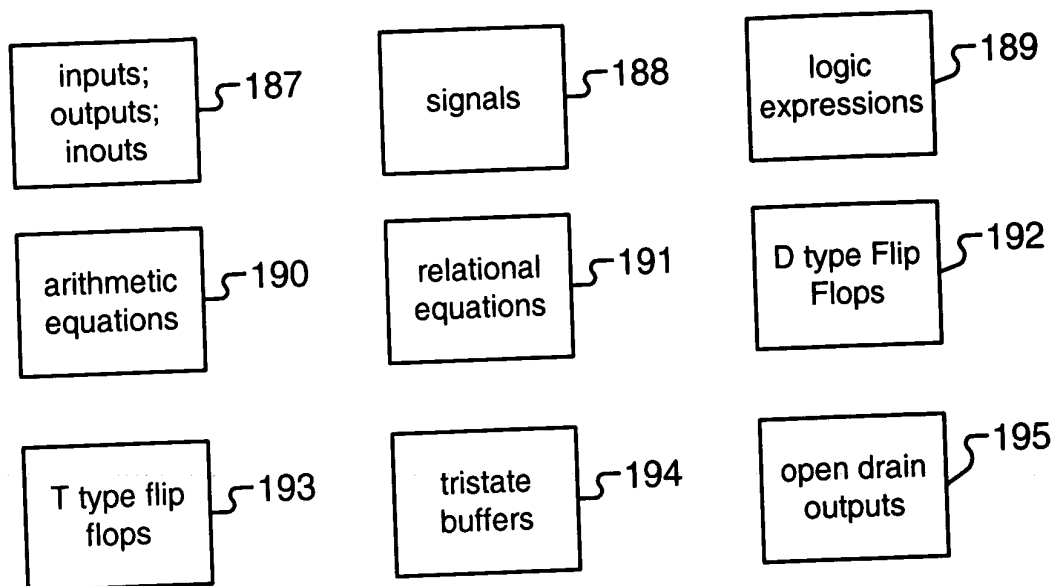


FIG. 5A

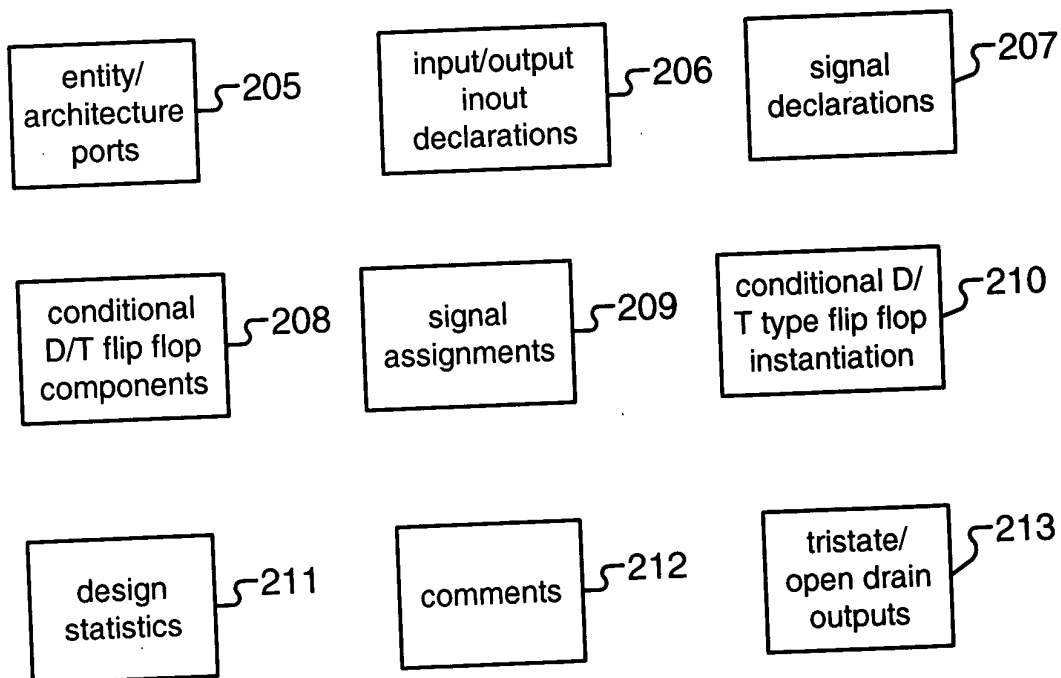


FIG. 6A

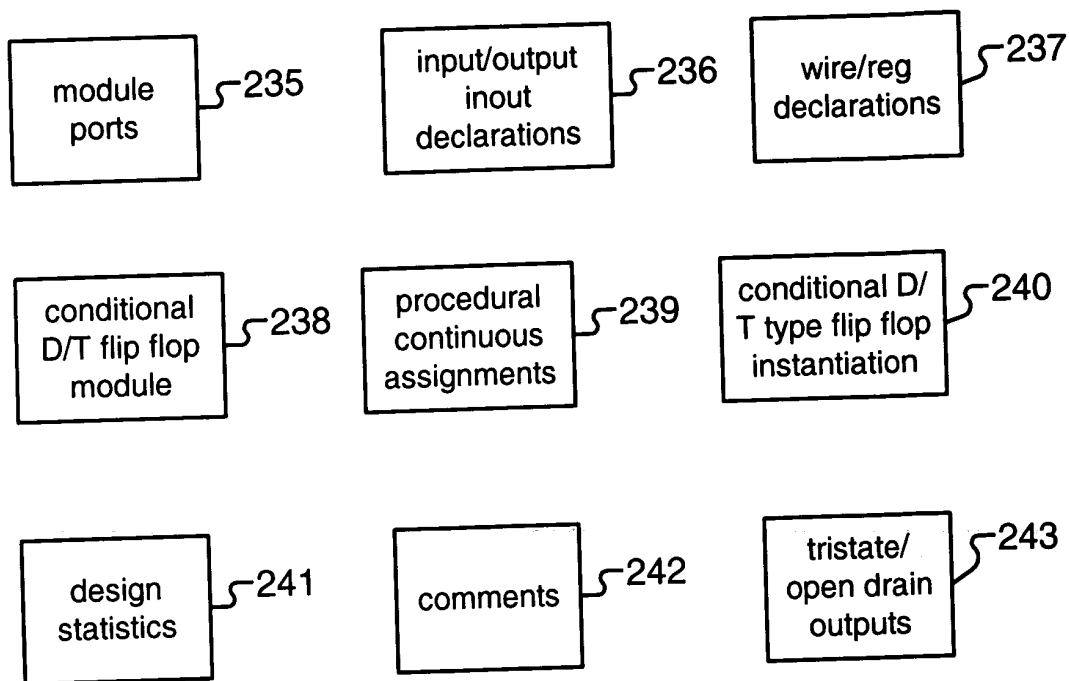


FIG. 6B

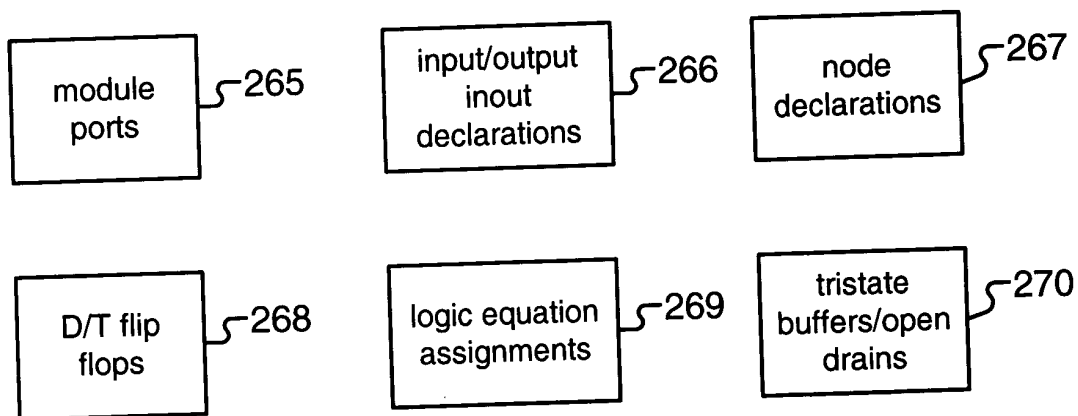


FIG. 6C

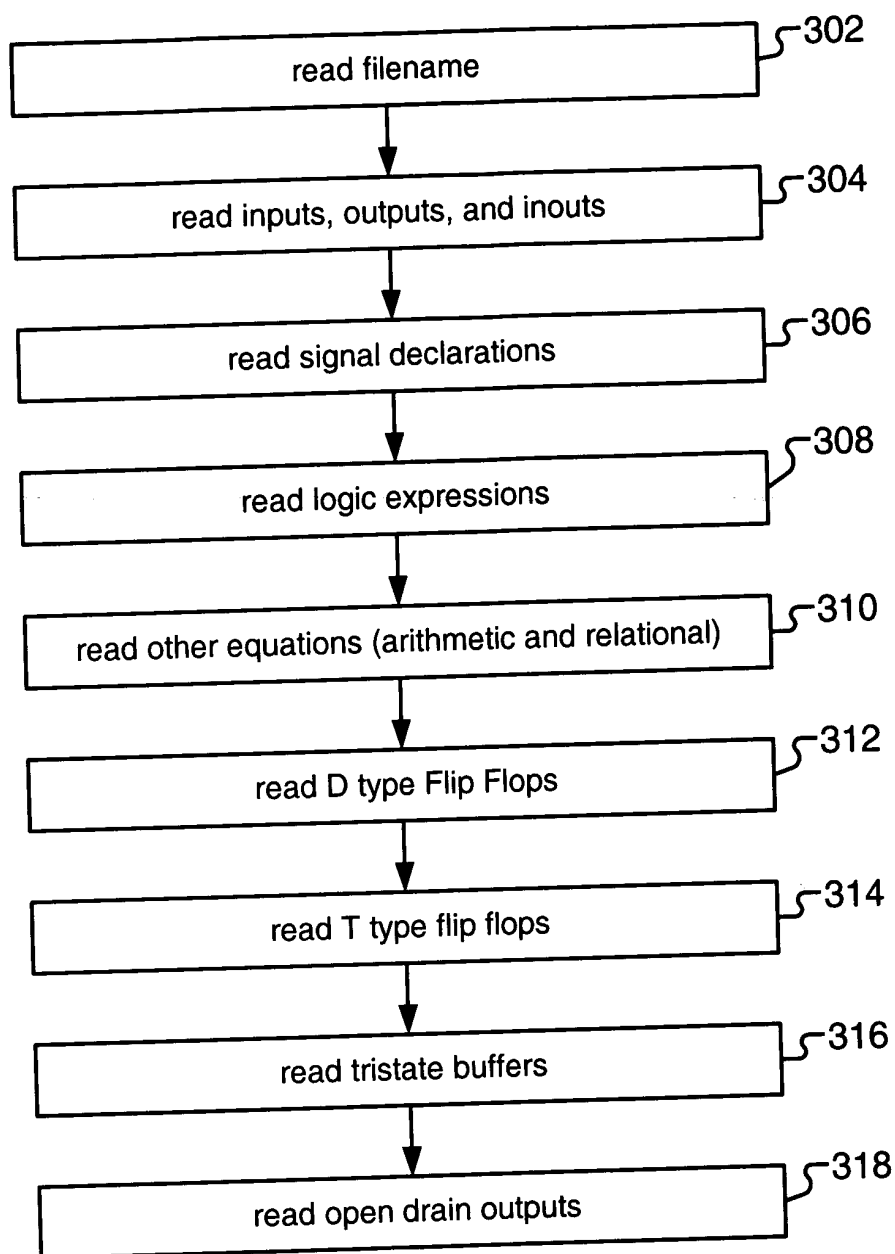
[illegible]

FIG. 5B

300

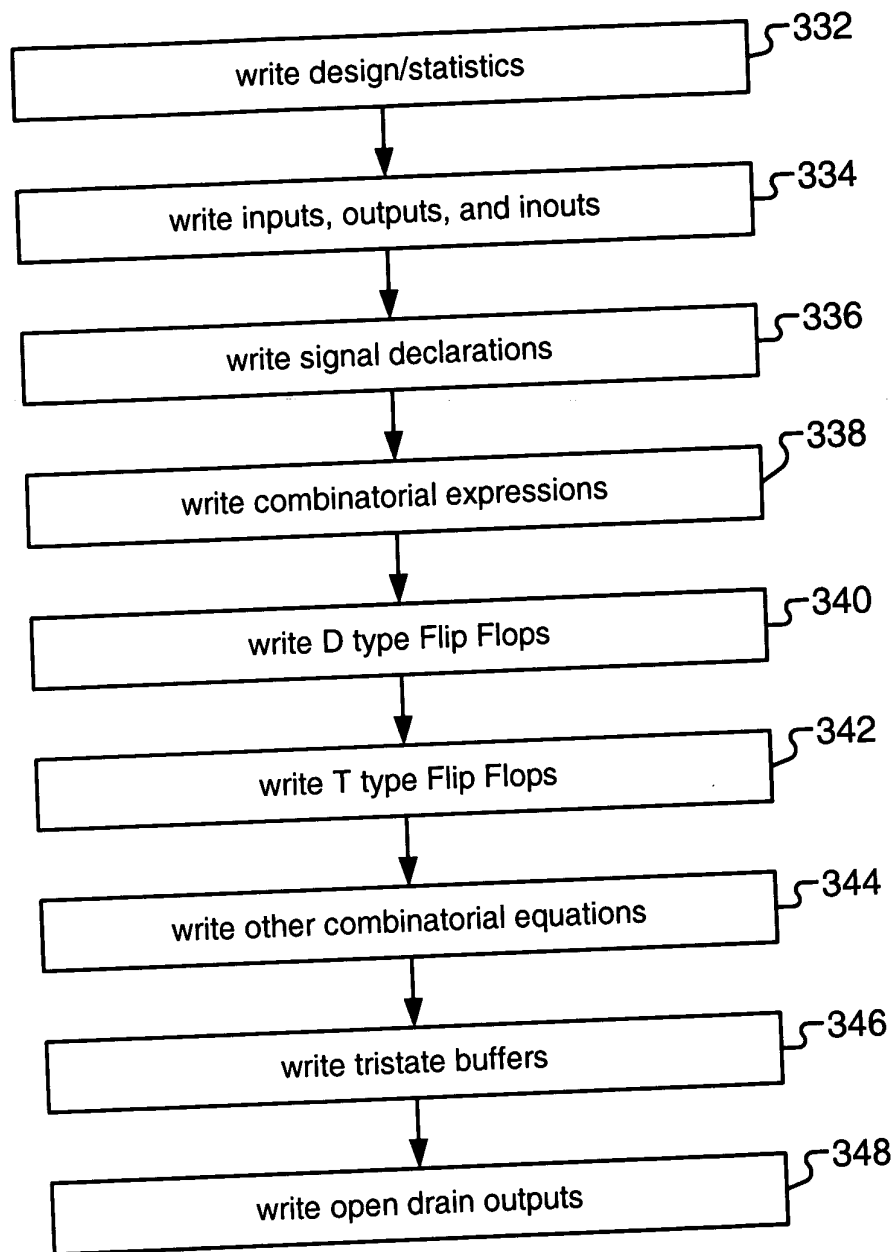


FIG. 7

Equations

SUBDESIGN 'counter4'

```
(  
    clk          : INPUT;  
    rst          : INPUT;  
    count0       : OUTPUT;  
    count1       : OUTPUT;  
    count2       : OUTPUT;  
    count3       : OUTPUT;  
)
```

VARIABLE

```
    _EQ001       : NODE;  
    _EQ002       : NODE;
```

BEGIN

```
    count0 = TFFE( VCC, GLOBAL( clk), !rst, VCC, VCC);  
    count1 = TFFE( count0, GLOBAL( clk), !rst, VCC, VCC);  
    count2 = TFFE( _EQ001, GLOBAL( clk), !rst, VCC, VCC);  
    _EQ001 = count0 & count1;  
    count3 = TFFE( _EQ002, GLOBAL( clk), !rst, VCC, VCC);  
    _EQ002 = count0 & count1 & count2;
```

END;

Fig. 8A

09847032-043001


```
-- Design name: counter4

-- Design Statistics
-- Number of Inputs      : 2
-- Number of Outs/Inouts : 4
-- Number of TFFEs       : 4
-- Number of EQ Equations : 2

library ieee;
use ieee.std_logic_1164.all;

entity tffe is
port (q : inout std_logic;
      t : in std_logic;
      clk : in std_logic;
      rst : in std_logic;
      pre : in std_logic;
      ce : in std_logic
);
end tffe;

architecture v1 of tffe is
signal d: std_logic;
begin

process (rst,pre,clk)
begin
if rst = '0' then
q <= '0';
elsif pre = '0' then
q <= '1';
elsif clk'event and clk = '1' then
if ce = '1' then
q <= d ;
end if ;
end if ;
end process ;

d <= t xor q;

end v1;

library ieee;
use ieee.std_logic_1164.all;
```

Fig. 8B-1

```

entity counter4 is
port (
clk : in std_logic;
rst : in std_logic;
count0 : inout std_logic;
count1 : inout std_logic;
count2 : inout std_logic;
count3 : inout std_logic);
end counter4 ;

```

architecture conversion of counter4 is

```

component tffe
port (q : inout std_logic;
      t : in std_logic;
      clk : in std_logic;
      rst : in std_logic;
      pre : in std_logic;
      ce : in std_logic
);
end component;

signal EQ001 : std_logic;
signal EQ002 : std_logic;

begin

-- 4 TFFE

tffe0 : tffe port map (count0,'1',clk,not rst,'1','1');
tffe1 : tffe port map (count1,count0,clk,not rst,'1','1');
tffe2 : tffe port map (count2,EQ001,clk,not rst,'1','1');
tffe3 : tffe port map (count3,EQ002,clk,not rst,'1','1');

-- 2 Comb Eqns(s)

EQ001 <= (count0 and count1);
EQ002 <= (count0 and count1 and count2);

-- 0 X Comb Eqn(s)

-- Additional Combinatorial Eqns

-- Assignments for equations w active low LHS

end conversion;

```

Fig. 8B-2

```

/* Design statistics

    Number of Inputs      : 2
    Number of Outputs     : 4
    Number of TFFEs       : 4
    Number of EQ Equations : 2
*/

module dffe (q,d,clk,rst,pre,ce);
output q;
input d,clk,rst,pre,ce;
reg q;

always @(posedge clk or negedge rst or negedge pre)
begin
    if (~rst)
        q = 1'b0;
    else if (~pre)
        q = 1'b1;
    else if (ce)
        q = d;
    end

endmodule

module tffe (q,t,clk,rst,pre,ce);
output q;
input t,clk,rst,pre,ce;
wire d;

dffe dffe0 (q,d,clk,rst,pre,ce);

assign d = t ^ q;

endmodule

```

Fig. 8C-1

```

module counter4 (
    clk,
    rst,
    count0,
    count1,
    count2,
    count3);

// 2 Inputs

input clk;
input rst;

// 4 Outputs

output count0;
output count1;
output count2;
output count3;

// 0 reg statements

// 1 wire statements

wire _EQ001;
wire _EQ002;

// 4 TFFE

tffe tffe0 (count0,1'b1,clk,!rst,1'b1,1'b1);
tffe tffe1 (count1,count0,clk,!rst,1'b1,1'b1);
tffe tffe2 (count2,_EQ001,clk,!rst,1'b1,1'b1);
tffe tffe3 (count3,_EQ002,clk,!rst,1'b1,1'b1);

// 2 Comb Eqns(s)

assign _EQ001 = count0 & count1;
assign _EQ002 = count0 & count1 & count2;

// 0 _X Comb Eqns(s)

endmodule

```

Fig. 8C-2

09847032 043001

" Conversion of counter4.tdo to counter4.abl

```
clk          pin;
rst          pin;
count0       pin;
count1       pin;
count2       pin;
count3       pin;

_EQ001       NODE;
_EQ002       NODE;
```

equations

```
count0.t = 1 ;
count0.clk = ( clk) ;
count0.ar = !!rst ;
"count0.ap = 1 ;
"count0.ce = 1 ;

count1.t = count0 ;
count1.clk = ( clk) ;
count1.ar = !!rst ;
"count1.ap = 1 ;
"count1.ce = 1 ;

count2.t = _EQ001 ;
count2.clk = ( clk) ;
count2.ar = !!rst ;
"count2.ap = 1 ;
"count2.ce = 1 ;
_EQ001 = count0 & count1;

count3.t = _EQ002 ;
count3.clk = ( clk) ;
count3.ar = !!rst ;
"count3.ap = 1 ;
"count3.ce = 1 ;
_EQ002 = count0 & count1 & count2;

END;
```

Fig. 8D

09847032.043001

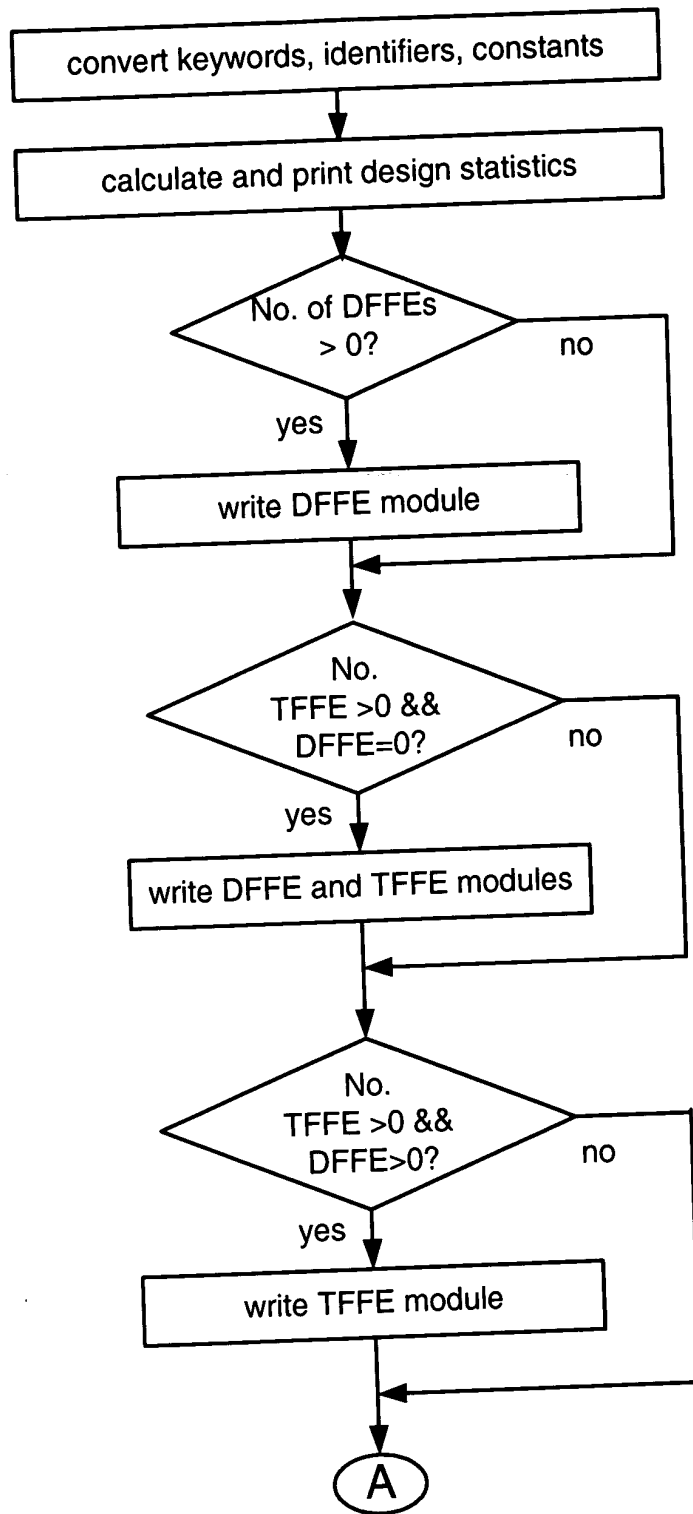


FIG. 9A

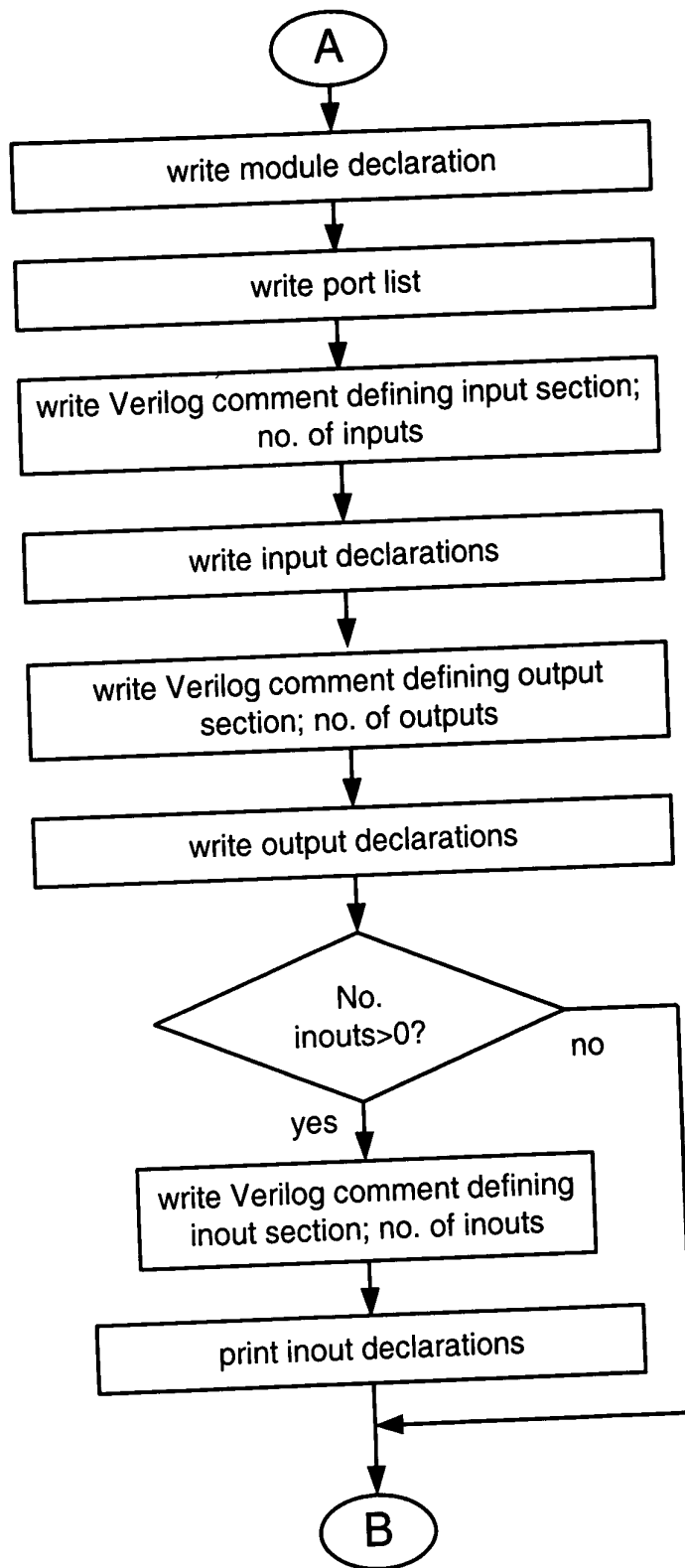


FIG. 9B

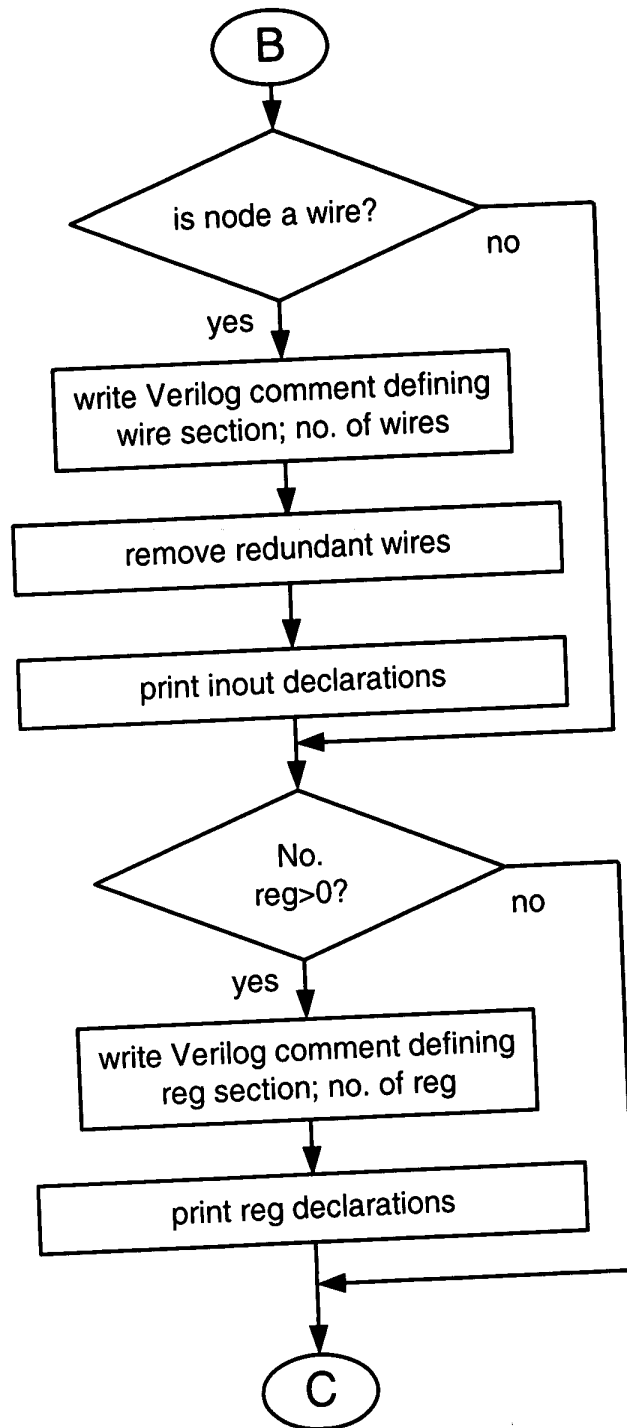


FIG. 9C

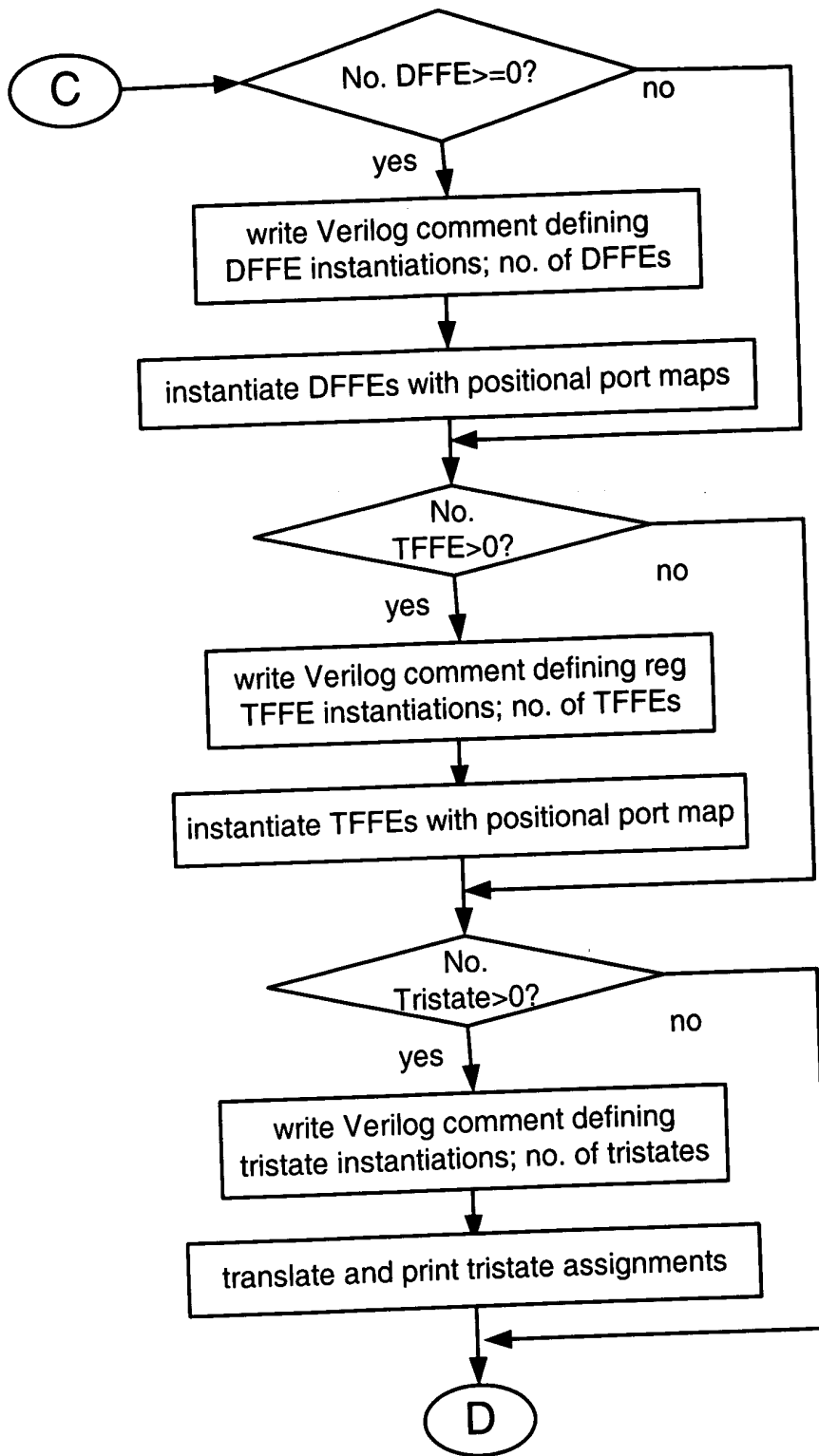


FIG. 9D

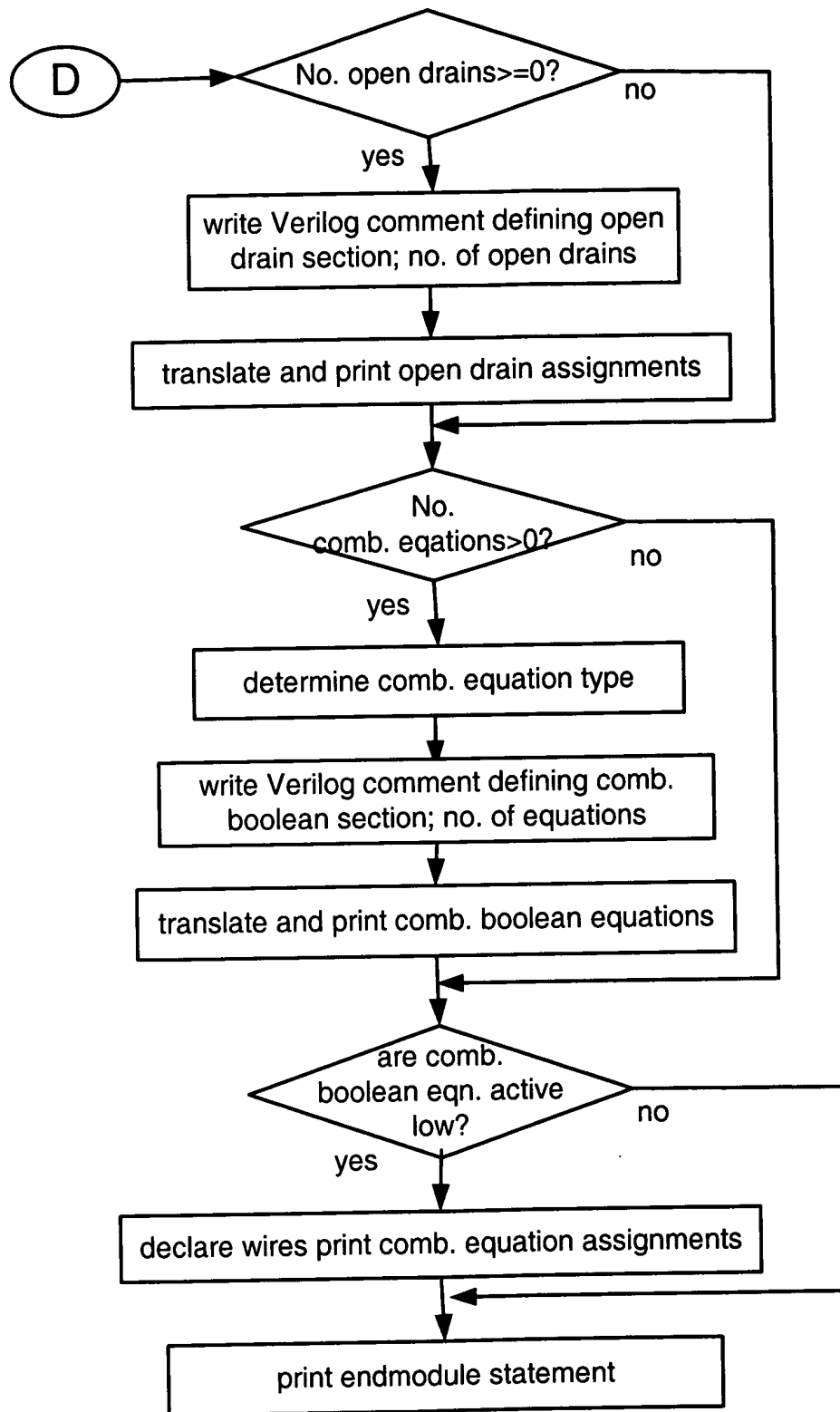


FIG. 9E